復号鍵漏洩耐性を持つ 鍵失効機能付きIDベース暗号の一般的な構成

高安 敦[1,2]　勝又 秀一[1,2]　松田 隆宏[2]

概要：ID ベース暗号を効果的に実利用するために，悪意のあるユーザーをシステムから除外する鍵失効機能は必須要件である．Boldyreva ら (ACM CCS'08) は，鍵失効機能を持つ ID ベース暗号 (RIBE) を初めて提案した．さらに，Seo と Emura (PKC'13) は，現実的に起こりうる人為的過失を考慮に入れ，復号鍵漏洩耐性 (DKER) を持つ RIBE 方式を初めて提案した．Seo と Emura の構成では，DKER を達成するためには RIBE の秘密鍵がリランダマイズ可能であることが本質的であり，これに倣い，ペアリングを用いてこれまで数多くの方式が提案されてきた．だが，格子に基づく RIBE では，秘密鍵をリランダマイズすることが困難であるため，DKER を持つ格子 RIBE がこれまで提案されていない．本稿で我々は，DKER を持つ RIBE の一般的な構成法を示す．より正確には，DKER を持つ RIBE は，DKER を持たない RIBE と 2 階層の階層型 IBE (HIBE) を組み合わせることで，一般的に構成できることを示す．提案構成法は，一般的構成であること・従来のように RIBE に秘密鍵リランダマイズを要求しないという意味で理論的に興味深い結果である．また，この構成によって初めて得られる DKER を持つ格子 RIBE は，DKER を持たない方式とほぼ同等の効率であるという点で，実用的にも意義深い結果である．

キーワード：鍵失効機能，ID ベース暗号，鍵漏洩耐性，格子暗号

Generic Construction of Revocable Identity-based Encryption with Decryption Key Exposure Resistance

Atsushi Takayasu[1,2]　Shuichi Katsumata[1,2]　Takahiro Matsuda[2]

Abstract: To use an identity-based encryption in a real system, the key revocation mechanism is required to omit malicious users from the system. Boldyreva et al. (ACM CCS'08) proposed the first revocable identity-based encryption (RIBE) scheme based on pairing. Subsequently, Seo and Emura (PKC'13) proposed a paring-based RIBE scheme in the more realistic security model that captures decryption key exposure resistance (DKER). However, there has been no lattice-based RIBE scheme with DKER, since the techniques used by Seo and Emura are paring specific, i.e., rerandomization of the secret keys. In this paper, we propose a generic construction of RIBE with DKER. Our result is theoretically interesting since the construction is generic and we do not require any further property on the underlying RIBE.Furthermore, the construction provides a practically important result; the first lattice-based RIBE with DKER, and the scheme is comparably efficient as a lattice-based RIBE scheme without DKER.

Keywords: revocable identity-based encryption, key exposure resistance, lattice-based cryptography

1. 序論

背景． ID ベース暗号 (identity-based encryption, IBE) は，鍵生成者 (key generation center, KGC) によって作成され

[1] 東京大学
The University of Tokyo
[2] 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)

た同一の公開パラメータを用いて，システムに登録された任意の受信者に対する暗号文を作成することができる．その暗号文は，メールアドレスなど，受信者の識別子 (id) に紐づいており，この id は任意の文字列が許される．そのため，通常の公開鍵暗号とは異なり，暗号化のための公開情報は，ひとたび入手すれば全ての受信者に対して暗号文を作成することができる．各受信者は，KGC から自分の id に紐づいた秘密鍵をもらい，自分の id に対する暗号文のみを復号することができる．IBE は Boneh と Franklin [3] のペアリングを用いた提案以来，これまで様々な構成が知られているが，本稿では格子を用いた構成 [1], [4], [6] に注目する．格子を用いた方式は，ペアリング方式とは異なり，量子計算機への耐性が期待できる．NIST による耐量子暗号標準化計画などを契機に，近年の暗号研究では，格子における方式の注目度が非常に高まっている．

各受信者ごとに公開鍵を生成する必要がないというのは IBE の長所の一つではあるが，逆に言えば，IBE では，仮に悪意のある振る舞いをするユーザーがシステム内部にいた時にでも，KGC によって一度秘密鍵が作成されてしまえばシステムから削除することができない．そのため，Boneh と Franklin は，時刻 t を導入し，各時刻ごとに鍵を作り直す対抗策を考えたが，この改変は効率を大幅に悪化させるものであった．Boldyreva ら [2] は，各時刻ごとに送信する情報を削減した，鍵失効機能を持つ IBE (revocable IBE, RIBE) 方式を初めて提案した．Boldyreva らの RIBE 構成では，暗号文は受信者の id と時刻 t に紐づいており，簡単に言えば $(\mathsf{id}, \mathsf{t})$ に対する 2 階層の階層型 IBE (hierarchical IBE, HIBE) の暗号文になっている．だが，受信者の秘密鍵は id のみに紐づいており，この秘密鍵を用いても暗号文を復号することができない．これは鍵失効機能のための必要条件であり，暗号文は 2 階層 HIBE の暗号文になっているが，秘密鍵を HIBE の秘密鍵とすることはできない．正規の受信者の復号を可能にするため，KGC は各時刻 t において，t に紐づいた更新鍵を公開する．そして，ユーザーは自分の秘密鍵と公開された更新鍵を用いて，$(\mathsf{id}, \mathsf{t})$ に紐づく復号鍵を作成する．この構成の思想は，2-out-of-2 fuzzy IBE と似通っており，暗号文は秘密鍵・更新鍵の片方のみでは復号できないが，それら二つにより作成される正当な復号鍵を用いれば復号可能である．ユーザーを二分木の葉によって管理する complete subtree 法 (CS 法) [12] を利用すると，最大ユーザー数を N とするとき送るべき更新鍵の数は $O(\log N)$ で済む．Boldyreva らの方式はペアリングを用いた selective 安全なものであったが，Libert と Vergnaud [11] はペアリングを用いた adaptive 安全な方式を提案した．Chen ら [5] は，格子で初めての RIBE を構成している．

Boldyreva らによる RIBE の提案後，この分野における最も大きな進展の一つは，Seo と Emura [13] による復号鍵漏洩耐性 (decryption key exposure resistance, DKER) を持つ RIBE 方式の提案である．Boldyreva らの方式では，システムユーザーの不注意等によって復号鍵が漏洩した際に，その受信者に対する暗号文の秘匿性が守られる保証はない．よって，このように現実的に起こりうるヒューマンエラーに対しては安全性が保証されていなかった．だが，Seo と Emura の方式は，ユーザー id^* の $\mathsf{t} \neq \mathsf{t}^*$ なる $(\mathsf{id}^*, \mathsf{t})$ に紐づく復号鍵がどれだけ漏洩しても，$(\mathsf{id}^*, \mathsf{t}^*)$ の暗号文の安全性が損なわれない．そのため，実用的に想定されるセキュリティ事故への対応が可能な重要な方式である．その後，DKER は RIBE 研究における基本的な安全性とされ，この安全性のもとで数多くの方式が提案されている [7], [8], [9], [10], [14], [15], [17].

DKER を持つ方式は全てペアリングを用いたもので，その構成はいずれも Seo と Emura のアプローチを踏襲している．このアプローチで核となるのは，RIBE の秘密鍵と更新鍵がリランダマイズできるという性質である．この性質は，$\mathsf{t} \neq \mathsf{t}^*$ なる $(\mathsf{id}^*, \mathsf{t})$ に紐づく復号鍵から $(\mathsf{id}^*, \mathsf{t}^*)$ の復号鍵を計算できないようにするために本質的なものだと思われる．ここで問題になるのは，格子 RIBE で DKER を満たす方式がいまだに提案されていないことである．現在知られている格子暗号のテクニックでは，RIBE のユーザー秘密鍵をリランダマイズをするためには，2 階層 HIBE の秘密鍵を渡す必要がある．そして，前述の通り，ユーザー id に HIBE の秘密鍵を渡してしまうと，ユーザーは t に関する更新鍵なしで，$(\mathsf{id}, \mathsf{t})$ の復号鍵を計算することができてしまうため，鍵失効機能を維持することができない．この問題を解決すべく，Takayasu と Watanabe [16] は，鍵リランダマイズを要求しない新たな方針のもとで DKER を満たす格子 RIBE の構成を目指した．だが，彼らの構成では，事前に設定された回数以内の復号鍵漏洩までしか安全性が保証されない．そのため，RIBE 研究の文脈において，鍵リランダマイズを要求しない新たな DKER の達成法は，理論的に重要な未解決問題であり，DKER を持つ RIBE 方式の格子での構成は，量子計算機完成後の IBE の実利用のために解決すべき重要な課題である．

成果．本稿で，我々は，鍵リランダマイズなしで DKER を達成する新たな RIBE の構成法を提案する．この構成法は，従来のように RIBE の秘密鍵がリランダマイズ可能であること要求しないという点で理論的に興味深い結果である．また，この構成法に基づいて我々が提案する初めての DKER を持つ格子 RIBE 方式は，DKER を持たない方式とほぼ同等の効率を達成しているという点で，実用的にも非常に意義深い．この方式は，耐量子性と DKER を併せ持つ初めての RIBE である．既存方式との効率の比較は，表 1 を参照．λ は安全性パラメータ，N は RIBE システムに登録できる最大ユーザー数である．

表 1: 格子 RIBE の比較（selective 安全）

方式	$\lvert\mathsf{PP}\rvert$	$\lvert\mathsf{ct}\rvert$	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ku}\rvert$	$\lvert\mathsf{dk}\rvert$	q	DKER
CLL+12 方式 [5]	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2)$	なし
TW17 方式 [16]	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda)$	$\tilde{O}(Q^2\lambda \log N)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	$\tilde{O}(Q\lambda^2)$	Q-bouded
提案方式	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^2) + \tilde{O}(\lambda \log N)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda^3)$	あり

従来技術的に難しいと考えられていた問題に対する我々の解決法は，非常に簡潔である．前述の通り，暗号文を $(\mathsf{id}, \mathsf{t})$ に対する HIBE 暗号文としたとき，鍵失効機能を有するためには，更新鍵なしでは復号できないようにするため，RIBE 秘密鍵を HIBE 秘密鍵とすることはできない．だが，DKER を達成するために，言い換えると，$\mathsf{t} \neq \mathsf{t}^*$ なる $(\mathsf{id}^*, \mathsf{t})$ の復号鍵から $(\mathsf{id}^*, \mathsf{t}^*)$ の復号鍵を導出できないようにするために，秘密鍵をリランダマイズさせようとすると，格子 RIBE では HIBE 秘密鍵を用いなければならないことを述べた．これを受けて，一見すると奇妙な構成に見えるかもしれないが，我々は，DKER を持たない RIBE は，2 階層 HIBE と組み合わせることで，DKER を持つ RIBE に変換可能であることを示す．我々の構成において，平文 M に対する DKER を持つ RIBE の暗号文は，平文をランダムに $\mathsf{M} = \mathsf{M}_1 \oplus \mathsf{M}_2$ と分解し，平文 M_1 に対する DKER を持たない RIBE 暗号文と，平文 M_2 に対する HIBE 暗号文からなる．この簡潔な構成は，DKER を持たない RIBE と 2 階層 HIBE の特徴を相補的に利用することができる．そこで，DKER なし RIBE の性質より，鍵失効機能を持つという点で，攻撃者は M_1 を得ることができない．また，HIBE の性質より，攻撃者は M_2 を得ることができない．これは，DKER が要求する $\mathsf{t} \neq \mathsf{t}^*$ なる $(\mathsf{id}^*, \mathsf{t})$ の復号鍵から $(\mathsf{id}^*, \mathsf{t}^*)$ の復号鍵を導出できないという条件が，まさに HIBE の安全性により保証されるからである．よって，この構成により，我々は，RIBE 研究の文脈において，DKER を達成するための非常に重要な結果を得ることができた．

2. 準備

2.1 RIBE

この節で，RIBE を定義する．

鍵失効操作． 本稿では，時刻 t における鍵失効者リスト RL_t を ID 空間 $\mathcal{ID}$ の部分集合と考える．さらに，従来の論文と異なるが，我々は RIBE の syntax に revoke アルゴリズムを含めない．なぜなら，revoke アルゴリズムは単に鍵失効者の ID をリスト RL_t に加える操作 $\mathsf{RL}_\mathsf{t} \leftarrow \mathsf{RL}_\mathsf{t} \cup \{\mathsf{id}\}$ であり，方式に依存しないためである．

Syntax． RIBE 方式 Π は，以下の 6 つのアルゴリズム $(\mathsf{Setup}, \mathsf{GenSK}, \mathsf{KeyUp}, \mathsf{GenDK}, \mathsf{Encrypt}, \mathsf{Decrypt})$ からなる:

$\mathsf{Setup}(1^\lambda) \to (\mathsf{PP}, \mathsf{MSK})$ ：安全性パラメータ 1^λ を入力とし，公開パラメータ PP とマスター秘密鍵 MSK を出力する．ただし，平文空間 $\mathcal{M}$，時刻空間 $\mathcal{T}$ と ID 空間 $\mathcal{ID}$ は安全性パラメータ 1^λ によって決まり，それらの情報は PP に含まれているとする．

$\mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}) \to (\mathsf{sk}_\mathsf{id}, \mathsf{MSK}')$ ： 公開パラメータ PP，マスター秘密鍵 MSK と $\mathsf{id} \in \mathcal{ID}$ を入力とし，id に対応する秘密鍵 sk_id と更新されたマスター秘密鍵 MSK' を出力する．

$\mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}, \mathsf{MSK}, \mathsf{RL}_\mathsf{t}) \to (\mathsf{ku}_\mathsf{t}, \mathsf{MSK}')$ ：公開パラメータ PP，時刻 t，マスター秘密鍵 MSK と鍵失効者リスト $\mathsf{RL}_\mathsf{t} \subseteq \mathcal{ID}$ を入力とし，時刻 t の更新鍵 ku_t と更新されたマスター秘密鍵 MSK' を出力する．

$\mathsf{GenDK}(\mathsf{PP}, \mathsf{sk}_\mathsf{id}, \mathsf{ku}_\mathsf{t}) \to \mathsf{dk}_{\mathsf{id},\mathsf{t}}$ or $\perp$ ：公開パラメータ PP，秘密鍵 sk_id と更新鍵 ku_t を入力とし，$(\mathsf{id}, \mathsf{t})$ に対する復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}}$ を出力する．ただし，id の鍵がすでに失効しているときには $\perp$ を出力する．

$\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, \mathsf{M}) \to \mathsf{ct}_{\mathsf{id},\mathsf{t}}$ ：公開パラメータ PP，id，時刻 t と平文 M を入力とし，暗号文 $\mathsf{ct}_{\mathsf{id},\mathsf{t}}$ を出力する．

$\mathsf{Decrypt}(\mathsf{PP}, \mathsf{dk}_{\mathsf{id},\mathsf{t}}, \mathsf{ct}_{\mathsf{id},\mathsf{t}}) \to \mathsf{M}'$ ：公開パラメータ PP，復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}}$ と暗号文 $\mathsf{ct}_{\mathsf{id},\mathsf{t}}$ を入力とし，復号結果 M' を出力する．

正当性． $(\mathsf{id}, \mathsf{t})$ に対する暗号文 $\mathsf{ct}_{\mathsf{id},\mathsf{t}}$ は，id が時刻 t までに鍵失効されていなければ，復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}}$ で正当に復号されなければならない．この制約を満たすため，全ての復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}}$ 生成の状況を想定する．全ての $\lambda \in \mathbb{N}$, $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{id} \in \mathcal{ID}$, $\mathsf{t} \in \mathcal{T}$, $\mathsf{M} \in \mathcal{M}$, $\mathsf{RL}_\mathsf{t} \subseteq \mathcal{ID} \setminus \{\mathsf{id}\}$ に対して，以下の操作の後 $\mathsf{M}' = \mathsf{M}$ が成立することを要求する：(1) $(\mathsf{sk}_\mathsf{id}, \mathsf{MSK}) \leftarrow \mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id})$. (2) $(\mathsf{ku}_\mathsf{t}, \mathsf{MSK}) \leftarrow \mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}, \mathsf{MSK}, \mathsf{RL}_\mathsf{t})$. (3) $\mathsf{dk}_{\mathsf{id},\mathsf{t}} \leftarrow \mathsf{GenDK}(\mathsf{PP}, \mathsf{sk}_\mathsf{id}, \mathsf{ku}_\mathsf{t})$. (4) $\mathsf{ct}_{\mathsf{id},\mathsf{t}} \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, \mathsf{M})$. (5) $\mathsf{M}' \leftarrow \mathsf{Decrypt}(\mathsf{PP}, \mathsf{dk}_{\mathsf{id},\mathsf{t}}, \mathsf{ct}_{\mathsf{id},\mathsf{t}})$.

安全性． RIBE 方式 $\Pi = (\mathsf{Setup}, \mathsf{GenSK}, \mathsf{KeyUp}, \mathsf{GenDK}, \mathsf{Encrypt}, \mathsf{Decrypt})$ の安全性を示す．紙面の都合上，本稿では selective 安全性のみを定義するが，adaptive 安全性はそこから自然に定義されるものである．RIBE は，DKER を満たすものが自然であるため，それを通常の安全性として定義する．その後に，DKER を満たさないものを弱 selective 安全性として定義する．

RIBE の（DKER を持つ）selective 安全性は，下記の攻撃者 $\mathcal{A}$ とチャレンジャー $\mathcal{C}$ によるゲームによって定義される．このゲームは，$\mathsf{t}_\mathsf{cu} = 1$ で初期化され，現在時刻を示すカウンターである．現実世界の設定を捉えるため，このカウンターによって攻撃者 $\mathcal{A}$ のクエリは制限される．ゲームは次のように進行する：

まず，攻撃者 $\mathcal{A}$ は，$(\mathsf{id}^*, \mathsf{t}^*) \in \mathcal{ID} \times \mathcal{T}$ をチャレンジャー $\mathcal{C}$ に送る．[*1]チャレンジャー $\mathcal{C}$ は $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$ を実行し，最初に $(\mathsf{kgc}, \mathsf{MSK})$ が入っているリスト $\mathtt{SKList}$ を作る．そして，攻撃者 $\mathcal{A}$ のクエリにしたがって作った $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}})$ をリスト $\mathtt{SKList}$ に加えていく．以下では，明記せずとも $\mathsf{id} \in \mathcal{ID}$ に対して秘密鍵を作ったときには $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}})$ をリスト $\mathtt{SKList}$ に加えていることとする．さらに，最初の時刻 $\mathsf{t}_{\mathsf{cu}} = 1$ において更新鍵を生成し $(\mathsf{ku}_1, \mathsf{MSK}') \leftarrow \mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}_{\mathsf{cu}} = 1, \mathsf{MSK}, \mathsf{RL}_1 = \emptyset)$，チャレンジャー $\mathcal{C}$ は，公開パラメータ PP と更新鍵 ku_1 を攻撃者 $\mathcal{A}$ に送る．そして，攻撃者 $\mathcal{A}$ は，以下の 4 つのクエリをチャレンジャー $\mathcal{C}$ に適応的に行う：

秘密鍵生成クエリ： 攻撃者 $\mathcal{A}$ のクエリ $\mathsf{id} \in \mathcal{ID}$ に対して，チャレンジャー $\mathcal{C}$ は $(\mathsf{id}, \star) \in \mathtt{SKList}$ かどうかを確認し，もしそうならば $\perp$ を攻撃者 $\mathcal{A}$ に送る．そうでなければ，チャレンジャー $\mathcal{C}$ は $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id})$ を実行する．ただし，攻撃者 $\mathcal{A}$ には何も送らない．[*2]

秘密鍵漏洩クエリ： 攻撃者 $\mathcal{A}$ のクエリ $\mathsf{id} \in \mathcal{ID}$ に対して，チャレンジャー $\mathcal{C}$ はある $\mathsf{sk}_{\mathsf{id}}$ に対して $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}) \in \mathtt{SKList}$ であることを確認し，秘密鍵 $\mathsf{sk}_{\mathsf{id}}$ を攻撃者 $\mathcal{A}$ に送る．そうでなければ $\perp$ を攻撃者 $\mathcal{A}$ に送る．

鍵失効 & 更新鍵クエリ： 攻撃者 $\mathcal{A}$ のクエリ $\Delta\mathsf{RL} \subseteq \mathcal{ID}$ に対して，チャレンジャー $\mathcal{C}$ は以下の条件が成り立つことを確認する：$\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \cap \Delta\mathsf{RL} = \emptyset$，全ての $\mathsf{id} \in \Delta\mathsf{RL}$ に対して $(\mathsf{id}, \star) \in \mathtt{SKList}$，$\mathsf{t}_{\mathsf{cu}} = \mathsf{t}^* - 1$ のとき，それまでに攻撃者 $\mathcal{A}$ が id^* の秘密鍵漏洩クエリを行ったならば，$\mathsf{id}^* \in \mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \cup \Delta\mathsf{RL}_{\mathsf{t}}$．[*3] もしこれらの条件が成り立たないなら，チャレンジャー $\mathcal{C}$ は攻撃者 $\mathcal{A}$ に $\perp$ を送る．そうでなければ，$\mathsf{t}_{\mathsf{cu}} \leftarrow \mathsf{t}_{\mathsf{cu}} + 1$ で現在時刻を進める．そして，チャレンジャー $\mathcal{C}$ は $\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \leftarrow \mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}-1} \cup \Delta\mathsf{RL}$ で鍵失効者リストを更新し，$(\mathsf{ku}_{\mathsf{t}_{\mathsf{cu}}}, \mathsf{MSK}') \leftarrow \mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}_{\mathsf{cu}}, \mathsf{MSK}, \mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}})$ を実行する．最後に，チャレンジャー $\mathcal{C}$ は現在時刻の更新鍵 $\mathsf{ku}_{\mathsf{t}_{\mathsf{cu}}}$ を攻撃者 $\mathcal{A}$ に送る．

復号鍵漏洩クエリ： 攻撃者 $\mathcal{A}$ のクエリ $(\mathsf{id}, \mathsf{t}) \in \mathcal{ID} \times \mathcal{T}$ に対して，チャレンジャー $\mathcal{C}$ は以下の条件が成り立つかを確認する：ある $\mathsf{sk}_{\mathsf{id}}$ に対して $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}) \in \mathtt{SKList}$，$\mathsf{t} \leq \mathsf{t}_{\mathsf{cu}}$，$(\mathsf{id}, \mathsf{t}) \neq (\mathsf{id}^*, \mathsf{t}^*)$．この条件を満たさなければ，チャレンジャー $\mathcal{C}$ は $\perp$ を攻撃者 $\mathcal{A}$ に送る．満たすならば，$\mathsf{GenDK}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}}, \mathsf{ku}_{\mathsf{t}})$ の出力を攻撃者 $\mathcal{A}$ に送る．

チャレンジクエリ： 攻撃者 $\mathcal{A}$ は一度だけこのクエリを行う．攻撃者 $\mathcal{A}$ の $|\mathsf{M}_0| = |\mathsf{M}_1|$ を満たすクエリ $(\mathsf{M}_0, \mathsf{M}_1)$ に対して，チャレンジャー $\mathcal{C}$ は一様ランダムに $b \in \{0,1\}$ を選び，$\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*} \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, \mathsf{t}^*, \mathsf{M}_b)$ を実行し，攻撃者 $\mathcal{A}$ にチャレンジ暗号文 $\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}$ を送る．

ある時点で，攻撃者 $\mathcal{A}$ は b の推定値 $b' \in \{0,1\}$ を出力し，ゲームを終了する．

このゲームにおける攻撃者 $\mathcal{A}$ の selective 安全性の優位性を，$\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda) := 2 \cdot |\Pr[b' = b] - 1/2|$ と定義する．

定義 1. 任意の確率的多項式時間攻撃者 $\mathcal{A}$ に対して優位性 $\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda)$ が安全性パラメータ 1^λ の *negligible* 関数であるとき，*RIBE* 方式 Π は *selective* 安全であるという．

DKER を持たない弱 selective 安全性は，上記の selective 安全性と同様に定義されるが，唯一の変更点は鍵失効 & 更新鍵クエリに以下の条件が加わることである．

- $\mathsf{t}_{\mathsf{cu}} = \mathsf{t}^* - 1$ のとき，それまでにある $\mathsf{t} \leq \mathsf{t}^* - 1$ に対して攻撃者 $\mathcal{A}$ が復号鍵漏洩クエリ $(\mathsf{id}^*, \mathsf{t})$ を行ったならば，$\mathsf{id}^* \in \mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \cup \Delta\mathsf{RL}_{\mathsf{t}}$．

この条件は，攻撃者 $\mathcal{A}$ が復号鍵 $\mathsf{dk}_{\mathsf{id}^*,\mathsf{t}}$ を持っているならば，id^* が時刻 t^* までに鍵失効することを保証するものである．つまり，通常の selective 安全性では，攻撃者 $\mathcal{A}$ は id^* の秘密鍵漏洩クエリを行わない限りその挙動に制約はないが，弱 selective 安全性では，加えて $(\mathsf{id}^*, \mathsf{t})$ の復号鍵漏洩クエリを行うならば，id^* を t^* までに鍵失効しなければならない．したがって，攻撃者 $\mathcal{A}$ が id^* の秘密鍵漏洩クエリを行うとき，二つの安全性は等価である．

弱 selective 安全性ゲームにおける攻撃者 $\mathcal{A}$ の優位性を，$\mathsf{Adv}^{\mathtt{RIBE\text{-}weak\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda)$ と定義する．

定義 2. 任意の確率的多項式時間攻撃者 $\mathcal{A}$ に対して優位性 $\mathsf{Adv}^{\mathtt{RIBE\text{-}weak\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda)$ が安全性パラメータ 1^λ の *negligible* 関数であるとき，*RIBE* 方式 Π は弱 *selective* 安全であるという．

2.2 2 階層 HIBE

この節で 2 階層 HIBE を定義する．ただし，本稿では通常の HIBE の一部の機能しか必要としないため，我々の一般的構成で必要となる機能だけを記す．具体的には，第 2 階層の $(\mathsf{id}_1, \mathsf{id}_2)$ の秘密鍵を KGC が作らない．よって，第 1 階層の id_1 の秘密鍵生成と，その秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ を用いる第 2 階層の $(\mathsf{id}_1, \mathsf{id}_2)$ の秘密鍵生成を明示的に GenSK と Delegate と分けて記述する．また，暗号文は第 2 階層のユーザー $(\mathsf{id}_1, \mathsf{id}_2)$ のみに作られるとする．

Syntax. 2 階層 HIBE 方式 Π は，以下の 5 つのアルゴリズム $(\mathsf{Setup}, \mathsf{GenSK}, \mathsf{Delegate}, \mathsf{Encrypt}, \mathsf{Decrypt})$ からなる：

$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{PP}, \mathsf{MSK})$ ：安全性パラメータ 1^λ を入力とし，公開パラメータ PP とマスター秘密鍵 MSK を出力する．ただし，平文空間 $\mathcal{M}$ と ID 空間 $\mathcal{ID}$ は安全性パラメータ 1^λ によって決まり，それらの情報は PP に含まれているとする．

[*1] selective 安全性と adaptive 安全性との違いは，いつチャレンジャー $\mathcal{C}$ に $(\mathsf{id}^*, \mathsf{t}^*) \in \mathcal{ID} \times \mathcal{T}$ を送るかのみである．これは次の節で紹介する 2 階層 HIBE においても同様である．

[*2] このクエリでは，秘密鍵 $\mathsf{sk}_{\mathsf{id}}$ を攻撃者 $\mathcal{A}$ には送らない．攻撃者 $\mathcal{A}$ が秘密鍵 $\mathsf{sk}_{\mathsf{id}}$ を受け取るのは，次の秘密鍵漏洩クエリによって行う．

[*3] この条件は，攻撃者 $\mathcal{A}$ が秘密鍵 $\mathsf{sk}_{\mathsf{id}^*}$ を持っているならば，id^* が時刻 t^* までに鍵失効することを保証するものである．

$\mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}_1) \to \mathsf{sk}_{\mathsf{id}_1}$ ：公開パラメータ PP，マスター秘密鍵 MSK と $\mathsf{id}_1 \in \mathcal{ID}$ を入力とし，id_1 に対応する秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ を出力する．

$\mathsf{Delegate}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2) \to \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ ：公開パラメータ PP，第 1 階層秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ と $\mathsf{id}_2 \in \mathcal{ID}$ を入力とし，$(\mathsf{id}_1, \mathsf{id}_2)$ に対応する秘密鍵 $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ を出力する．

$\mathsf{Encrypt}(\mathsf{PP}, (\mathsf{id}_1, \mathsf{id}_2), \mathsf{M}) \to \mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2}$ ：公開パラメータ PP，$(\mathsf{id}_1, \mathsf{id}_2)$ と平文 M を入力とし，暗号文 $\mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2}$ を出力する．

$\mathsf{Decrypt}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}, \mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2}) \to \mathsf{M}'$ ：公開パラメータ PP，復号鍵 $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ と暗号文 $\mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2}$ を入力とし，復号結果 M' を出力する．

正当性． 全ての $\lambda \in \mathbb{N}$, $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$, $\mathsf{sk}_{\mathsf{id}_1} \leftarrow \mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}_1)$, $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2} \leftarrow \mathsf{Delegate}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2)$, $\mathsf{M} \in \mathcal{M}$, $\mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2} \leftarrow \mathsf{Encrypt}(\mathsf{PP}, (\mathsf{id}_1, \mathsf{id}_2), \mathsf{M})$ に対して $\mathsf{Decrypt}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}, \mathsf{ct}_{\mathsf{id}_1,\mathsf{id}_2}) = \mathsf{M}$ が成り立つことを要求する．

安全性． 2 階層 HIBE 方式 $\Pi = (\mathsf{Setup}, \mathsf{GenSK}, \mathsf{Delegate}, \mathsf{Encrypt}, \mathsf{Decrypt})$ の安全性を定義する．RIBE の場合と同様，selective 安全性のみを記す．HIBE の安全性は，攻撃者 $\mathcal{A}$ とチャレンジャー $\mathcal{C}$ による以下のゲームによって定義される．

まず，攻撃者 $\mathcal{A}$ は，$(\mathsf{id}_1^*, \mathsf{id}_2^*) \in (\mathcal{ID})^2$ をチャレンジャー $\mathcal{C}$ に送る．チャレンジャー $\mathcal{C}$ は $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$ を実行し，最初に $(\mathsf{kgc}, \mathsf{MSK})$ が入っているリスト SKList を作る．そして，攻撃者 $\mathcal{A}$ のクエリにしたがって作った $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1})$ または $((\mathsf{id}_1, \mathsf{id}_2), \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2})$ を SKList に加えていく．以下では，明記せずとも $\mathsf{id}_1 \in \mathcal{ID}$ または $(\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$ に対して秘密鍵を作ったときには $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1})$ または $((\mathsf{id}_1, \mathsf{id}_2), \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2})$ をリスト SKList に加えていることとする．次に，チャレンジャー $\mathcal{C}$ は，公開パラメータ PP を攻撃者 $\mathcal{A}$ に送る．そして，攻撃者 $\mathcal{A}$ は，以下の 4 つのクエリをチャレンジャー $\mathcal{C}$ に適応的に行う:

第 1 階層秘密鍵生成クエリ： 攻撃者 $\mathcal{A}$ のクエリ $\mathsf{id}_1 \in \mathcal{ID}$ に対して，チャレンジャー $\mathcal{C}$ は $(\mathsf{id}_1, *) \in$ SKList かどうかを確認し，もしそうならば $\perp$ を攻撃者 $\mathcal{A}$ に送る．そうでなければ，チャレンジャー $\mathcal{C}$ は $\mathsf{sk}_{\mathsf{id}_1} \leftarrow \mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}_1)$ を計算する．ただし，攻撃者 $\mathcal{A}$ には何も送らない．[*4]

第 1 階層秘密鍵漏洩クエリ： 攻撃者 $\mathcal{A}$ のクエリ $\mathsf{id}_1 \in \mathcal{ID}$ に対して，チャレンジャー $\mathcal{C}$ は $\mathsf{id}_1 \neq \mathsf{id}_1^*$，ある $\mathsf{sk}_{\mathsf{id}_1}$ に対して $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1}) \in$ SKList が成り立つことが確認し，秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ を攻撃者 $\mathcal{A}$ に送る．そうでなければ $\perp$ を攻撃者 $\mathcal{A}$ に送る．

[*4] このクエリでは，秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ を攻撃者 $\mathcal{A}$ には送らない．攻撃者 $\mathcal{A}$ が秘密鍵 $\mathsf{sk}_{\mathsf{id}_1}$ を受け取るのは，次の reveal クエリによって行う．この操作は，攻撃者 $\mathcal{A}$ が $\mathsf{id}_2 \neq \mathsf{id}_2^*$ なる第 2 階層の秘密鍵 $\mathsf{sk}_{\mathsf{id}_1^*,\mathsf{id}_2}$ を受け取れるように定義するためのものである．

第 2 階層秘密鍵漏洩クエリ： 攻撃者 $\mathcal{A}$ のクエリ $(\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$ に対して，チャレンジャー $\mathcal{C}$ は $(\mathsf{id}_1, \mathsf{id}_2) \neq (\mathsf{id}_1^*, \mathsf{id}_2^*)$，ある $\mathsf{sk}_{\mathsf{id}_1}$ に対して $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1}) \in$ SKList，また，$((\mathsf{id}_1, \mathsf{id}_2), \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}) \notin$ SKList であることを確認し，$\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2} \leftarrow \mathsf{Delegate}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2)$ を実行した後，攻撃者 $\mathcal{A}$ に秘密鍵 $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ を送る．そうでなければ $\perp$ を攻撃者 $\mathcal{A}$ に送る．

チャレンジクエリ： 攻撃者 $\mathcal{A}$ は一度だけこのクエリを行う．攻撃者 $\mathcal{A}$ の $|\mathsf{M}_0| = |\mathsf{M}_1|$ を満たすクエリ $(\mathsf{M}_0, \mathsf{M}_1)$ に対して，チャレンジャー $\mathcal{C}$ は一様ランダムに $b \in \{0, 1\}$ を選び，$\mathsf{ct}_{\mathsf{id}_1^*,\mathsf{id}_2^*} \leftarrow \mathsf{Encrypt}(\mathsf{PP}, (\mathsf{id}_1^*, \mathsf{id}_2^*), \mathsf{M}_b)$ を実行し，攻撃者 $\mathcal{A}$ にチャレンジ暗号文 $\mathsf{ct}_{\mathsf{id}_1^*,\mathsf{id}_2^*}$ を送る．

ある時点で，攻撃者 $\mathcal{A}$ は b の推定値 $b' \in \{0, 1\}$ を出力し，ゲームを終了する．

このゲームにおける攻撃者 $\mathcal{A}$ の selective 安全性の優位性を，$\mathsf{Adv}^{\mathtt{HIBE\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda) := 2 \cdot |\Pr[b' = b] - 1/2|$ と定義する．

定義 3. 任意の確率的多項式時間攻撃者に対して優位性 $\mathsf{Adv}^{\mathtt{HIBE\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda)$ が安全性パラメータ 1^λ の *negligible* 関数であるとき，2 階層 *HIBE* 方式 Π は *selective* 安全であるという．

3. DKER を持つ RIBE の一般的構成

この章で，DKER を持たない RIBE 方式と 2 階層 HIBE 方式を用いた DKER を持つ RIBE の一般的構成を記す．以下では，DKER を持たない RIBE 方式，2 階層 HIBE 方式のパラメータやアルゴリズムに関しては，例えば公開パラメータに関しては $\mathsf{r.PP}, \mathsf{h.PP}$ のように，r. と h. を前につけて表記する．

3.1 構成

$\mathsf{Setup}(1^\lambda) \to (\mathsf{PP}, \mathsf{MSK})$ ：安全性パラメータ 1^λ を入力とし，RIBE と HIBE の Setup アルゴリズム $\mathsf{r.Setup}(1^\lambda) \to (\mathsf{r.PP}, \mathsf{r.MSK}), \mathsf{h.Setup}(1^\lambda) \to (\mathsf{h.PP}, \mathsf{h.MSK})$ を実行し，公開パラメータ $\mathsf{PP} := (\mathsf{r.PP}, \mathsf{h.PP})$ とマスター秘密鍵 $\mathsf{MSK} := (\mathsf{r.MSK}, \mathsf{h.MSK})$ を出力する．ただし，平文空間[*5] $\mathcal{M}$，ID 空間 $\mathcal{ID}$ と時刻空間 $\mathcal{T}$ は安全性パラメータ 1^λ によって決まり，それぞれ $\mathcal{M} = \mathsf{r}.\mathcal{M} = \mathsf{h}.\mathcal{M}, \mathcal{ID} = \mathsf{r}.\mathcal{ID} \subseteq \mathsf{h}.\mathcal{ID}, \mathcal{T} = \mathsf{r}.\mathcal{T} \subseteq \mathsf{h}.\mathcal{ID}$ を満たし，それらの情報は PP に含まれているとする．

$\mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}) \to (\mathsf{sk}_{\mathsf{id}}, \mathsf{MSK}')$ ：公開パラメータ PP，マスター秘密鍵 MSK と $\mathsf{id} \in \mathcal{ID}$ を入力とし，RIBE と HIBE の GenSK アルゴリズム $\mathsf{r.GenSK}(\mathsf{r.PP}, \mathsf{r.MSK}, \mathsf{id}) \to (\mathsf{r.sk}_{\mathsf{id}}, \mathsf{r.MSK}'), \mathsf{h.GenSK}(\mathsf{h.PP}, \mathsf{h.MSK}, \mathsf{id}) \to \mathsf{h.sk}_{\mathsf{id}}$ を実行し，id に対する秘密鍵 $\mathsf{sk}_{\mathsf{id}} = (\mathsf{r.sk}_{\mathsf{id}}, \mathsf{h.sk}_{\mathsf{id}})$ と更新されたマスター秘密鍵 $\mathsf{MSK}' = (\mathsf{r.MSK}', \mathsf{h.MSK})$

[*5] 任意の $\mathsf{M}, \mathsf{M}' \in \mathcal{M}$ に対して，$\mathsf{M} \oplus \mathsf{M}' \in \mathcal{M}$ が成立するとする．

を出力する．

$\mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}, \mathsf{MSK}, \mathsf{RL_t}) \to (\mathsf{ku_t}, \mathsf{MSK}')$ ： 公開パラメータ PP，時刻 t，マスター秘密鍵 MSK，鍵失効者リスト $\mathsf{RL_t}$ を入力とし，RIBE の KeyUp アルゴリズム $\mathsf{r.KeyUp}(\mathsf{r.PP}, \mathsf{t}, \mathsf{r.MSK}, \mathsf{RL_t}) \to (\mathsf{r.ku_t}, \mathsf{r.MSK}')$ を実行し，時刻 t の更新鍵 $\mathsf{ku_t} := \mathsf{r.ku_t}$ と更新されたマスター秘密鍵 $\mathsf{MSK}' = (\mathsf{r.MSK}', \mathsf{h.MSK})$ を出力する．

$\mathsf{GenDK}(\mathsf{PP}, \mathsf{sk_{id}}, \mathsf{ku_t}) \to \mathsf{dk_{id,t}}$ or $\perp$ ：公開パラメータ PP，秘密鍵 $\mathsf{sk_{id}}$ と更新鍵 $\mathsf{ku_t}$ を入力とし，RIBE の GenDK アルゴリズム $\mathsf{r.GenDK}(\mathsf{r.PP}, \mathsf{r.sk_{id}}, \mathsf{ku_t})$ と HIBE の Delegate アルゴリズム $\mathsf{h.Delegate}(\mathsf{h.PP}, \mathsf{h.sk_{id}}, \mathsf{t}) \to \mathsf{h.sk_{id,t}}$ を実行し，$\mathsf{r.GenDK}(\mathsf{r.PP}, \mathsf{r.sk_{id}}, \mathsf{ku_t})$ の出力が $\perp$ ならば $\perp$ を出力する．そうでなければ，$\mathsf{dk_{id,t}} := (\mathsf{r.dk_{id,t}}, \mathsf{h.sk_{id,t}})$ を出力する．

$\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, \mathsf{M}) \to \mathsf{ct_{id,t}}$ ：公開パラメータ PP，$(\mathsf{id}, \mathsf{t})$ と平文 M を入力とし，まず，平文を $\mathsf{M} = \mathsf{r.M} \oplus \mathsf{h.M}$ という条件のもと $\mathsf{r.M}, \mathsf{h.M} \in \mathcal{M}$ をランダムにサンプルする．そして，それぞれの平文に対して RIBE と HIBE の Encrypt アルゴリズム $\mathsf{r.Encrypt}(\mathsf{r.PP}, \mathsf{id}, \mathsf{t}, \mathsf{r.M}) \to \mathsf{r.ct_{id,t}}$, $\mathsf{h.Encrypt}(\mathsf{h.PP}, (\mathsf{id}, \mathsf{t}), \mathsf{h.M}) \to \mathsf{h.ct_{id,t}}$ を実行し，暗号文 $\mathsf{ct_{id,t}} := (\mathsf{r.ct_{id,t}}, \mathsf{h.ct_{id,t}})$ を出力する．

$\mathsf{Decrypt}(\mathsf{PP}, \mathsf{dk_{id,t}}, \mathsf{ct_{id,t}}) \to \mathsf{M}'$ ：公開パラメータ PP，復号鍵 $\mathsf{dk_{id,t}} := (\mathsf{r.dk_{id,t}}, \mathsf{h.sk_{id,t}})$ と暗号文 $\mathsf{ct_{id,t}} = (\mathsf{r.ct_{id,t}}, \mathsf{h.ct_{id,t}})$ を入力とし，それぞれ RIBE と HIBE の復号アルゴリズム $\mathsf{r.Decrypt}(\mathsf{r.PP}, \mathsf{r.dk_{id,t}}, \mathsf{r.ct_{id,t}}) \to \mathsf{r.M}'$, $\mathsf{h.Decrypt}(\mathsf{h.PP}, \mathsf{h.sk_{id,t}}, \mathsf{h.ct_{id,t}}) \to \mathsf{h.M}'$ を実行し，$\mathsf{M}' = \mathsf{r.M}' \oplus \mathsf{h.M}'$ を復号結果として出力する．

正当性は，元となる RIBE 方式と 2 階層 HIBE 方式の正当性より成り立つ．

3.2 安全性

第 3.1 章で構成した RIBE 方式の安全性について，この章で以下の定理を証明する．

定理 1. *RIBE* 方式 $\mathsf{r.\Pi}$ が弱 *selective* 安全で，かつ，2 階層 *HIBE* 方式 $\mathsf{h.\Pi}$ が *selective* 安全であるならば，第 *3.1* 章の *RIBE* 方式 Π は *selective* 安全である．

本稿では紙面の都合上省略するが，元となる RIBE 方式が弱 adaptive 安全で 2 階層 HIBE 方式が adaptive 安全ならば，得られる RIBE 方式は adaptive 安全な方式となる．

Proof. RIBE 方式 Π の任意の確率的多項式時間攻撃者 $\mathcal{A}$ は，その挙動によって以下の 2 タイプに分けることができる：

Type-1: id^* の秘密鍵漏洩クエリを行う攻撃者 $\mathcal{A}_1$．

Type-2: id^* の秘密鍵漏洩クエリを行わない攻撃者 $\mathcal{A}_2$．

定義より，$\mathcal{A}_1$ と $\mathcal{A}_2$ は排他的，かつ，攻撃者 $\mathcal{A}$ の挙動を全て網羅している．よって，詳細は省略するが，$\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_1}(\lambda) + \mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_2}(\lambda)$ が成り立つ．定理 1 を証明するために，二つの関係 $\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathtt{RIBE\text{-}weak\text{-}sel}}_{\mathsf{r.\Pi},\mathsf{r.}\mathcal{A}}(\lambda)$ と $\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_2}(\lambda) = \mathsf{Adv}^{\mathtt{HIBE\text{-}sel}}_{\mathsf{h.\Pi},\mathsf{h.}\mathcal{A}}(\lambda)$ が成り立つことを示す．この証明の中では，直感的な理解のために，$\mathsf{ct_{id,t}}(\mathsf{M})$ のように，暗号文がどの平文に対するものかを明記する．

まず，一つ目の関係を示す．そのために，RIBE 方式 Π の selective 安全性を破る攻撃者 $\mathcal{A}_1$ を用いて，RIBE 方式 $\mathsf{r.\Pi}$ の弱 selective 安全性を破る攻撃者 $\mathsf{r.}\mathcal{A}$ が構成できることを示す．攻撃者 $\mathsf{r.}\mathcal{A}$ は，RIBE 方式 $\mathsf{r.\Pi}$ のチャレンジャー $\mathsf{r.}\mathcal{C}$ と弱 selective 安全性ゲームを行いながら，RIBE 方式 Π のチャレンジャーとして振る舞い，攻撃者 $\mathcal{A}_1$ と selective 安全性ゲームを行う．定義より，攻撃者 $\mathcal{A}_1$ は id^* の秘密鍵漏洩クエリをチャレンジャー $\mathcal{C}$ に行うが，第 2.1 章で述べた通り，このときには RIBE の selective 安全性ゲームと弱 selective 安全性ゲームにおける攻撃者の挙動が等しくなることに着目して証明を行う．

攻撃者 $\mathcal{A}_1$ から $(\mathsf{id}^*, \mathsf{t}^*) \in \mathcal{ID} \times \mathcal{T}$ を受け取ると，攻撃者 $\mathsf{r.}\mathcal{A}$ はそれを RIBE 方式 $\mathsf{r.\Pi}$ のチャレンジャー $\mathsf{r.}\mathcal{C}$ に送る．そして，チャレンジャー $\mathsf{r.}\mathcal{C}$ から公開パラメータ $\mathsf{r.PP}$ と更新鍵 $\mathsf{r.ku_1}$ を受け取る．次に，攻撃者 $\mathsf{r.}\mathcal{A}$ は $\mathsf{h.Setup}(1^\lambda) \to (\mathsf{h.PP}, \mathsf{h.MSK})$ を実行し，$(\mathsf{h.kgc}, \mathsf{h.MSK})$ が入っているリスト SKList を作り，公開パラメータ $\mathsf{PP} := (\mathsf{r.PP}, \mathsf{h.PP})$ と更新鍵 $\mathsf{ku_1} := \mathsf{r.ku_1}$ を攻撃者 $\mathcal{A}_1$ に送る．その後，攻撃者 $\mathsf{r.}\mathcal{A}$ は攻撃者 $\mathcal{A}$ の秘密鍵漏洩クエリ・鍵失効 & 更新鍵クエリ・復号鍵漏洩クエリに適切に答えられることを確認する．攻撃者 $\mathsf{r.}\mathcal{A}$ は，HIBE のマスター秘密鍵 $\mathsf{h.MSK}$ を持っているため，全ての id, t に対する HIBE 鍵生成が行える．また，攻撃者 $\mathcal{A}_1$ の挙動が RIBE の selective 安全性ゲームと弱 selective 安全性ゲームで等しいことから，攻撃者 $\mathcal{A}_1$ のクエリに対して，チャレンジャー $\mathsf{r.}\mathcal{C}$ に同じクエリを行うことで必要な RIBE の鍵を全て手に入れることができる．

攻撃者 $\mathcal{A}_1$ のチャレンジクエリ $(\mathsf{M}_0, \mathsf{M}_1)$ を受け取ると，攻撃者 $\mathsf{r.}\mathcal{A}$ は，ランダムに $\mathsf{h.M} \leftarrow \mathcal{M}$ をサンプルし，チャレンジャー $\mathsf{r.}\mathcal{C}$ へのチャレンジクエリ $(\mathsf{M}_0 \oplus \mathsf{h.M}, \mathsf{M}_1 \oplus \mathsf{h.M})$ を行い，チャレンジ暗号文 $\mathsf{r.ct_{id^*,t^*}}(\mathsf{M}_b \oplus \mathsf{h.M})$ を受け取る．そして，$\mathsf{h.ct_{id^*,t^*}}(\mathsf{h.M}) \leftarrow \mathsf{h.Encrypt}(\mathsf{h.PP}, (\mathsf{id}^*, \mathsf{t}^*), \mathsf{h.M})$ を実行し，チャレンジ暗号文 $\mathsf{ct_{id^*,t^*}}(\mathsf{M}_b) = (\mathsf{r.ct_{id^*,t^*}}(\mathsf{M}_b \oplus \mathsf{h.M}), \mathsf{h.ct_{id^*,t^*}}(\mathsf{h.M}))$ を攻撃者 $\mathcal{A}_1$ に送る．このチャレンジ暗号文は，実際の方式と同じ分布に従う．つまり，$\mathsf{M}_b \oplus \mathsf{h.M}$ と $\mathsf{h.M}$ は，$(\mathsf{M}_b \oplus \mathsf{h.M}) \oplus \mathsf{h.M} = \mathsf{M}_b$ を満たす一様ランダムな分布に従う．いずれの $b \in \{0, 1\}$ に対しても，ゲームの最後に，攻撃者 $\mathcal{A}_1$ から b' 受け取ると，同じ b' をチャレンジャー $\mathsf{r.}\mathcal{C}$ に送りゲームを終了する．この構成によって，最初に述べたように RIBE 方式 $\mathsf{r.\Pi}$ の弱 selective 安全性を破る攻撃者 $\mathsf{r.}\mathcal{A}$ が構成でき，$\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_1}(\lambda) = \mathsf{Adv}^{\mathtt{RIBE\text{-}weak\text{-}sel}}_{\mathsf{r.\Pi},\mathsf{r.}\mathcal{A}}(\lambda)$ が成り立つ．

次に，RIBE 方式 Π の selective 安全性を破る攻撃者 $\mathcal{A}_2$ を用いて，HIBE 方式 $\mathsf{h}.\Pi$ の selective 安全性を破る攻撃者 $\mathsf{h}.\mathcal{A}$ を構成する．そして，このとき，二つ目の関係 $\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_2}(\lambda) = \mathsf{Adv}^{\mathtt{HIBE\text{-}sel}}_{\mathsf{h}.\Pi,\mathsf{h}.\mathcal{A}}(\lambda)$ が成り立つことを示す．攻撃者 $\mathsf{h}.\mathcal{A}$ は，HIBE 方式 $\mathsf{h}.\Pi$ のチャレンジャー $\mathsf{h}.\mathcal{C}$ と selective 安全性ゲームを行いながら，RIBE 方式 Π のチャレンジャーとして振る舞い，攻撃者 $\mathcal{A}_2$ と selective 安全性ゲームを行う．

攻撃者 $\mathcal{A}_2$ から $(\mathsf{id}^*, \mathsf{t}^*) \in \mathcal{ID} \times \mathcal{T}$ を受け取ると，攻撃者 $\mathsf{h}.\mathcal{A}$ はそれを HIBE 方式 $\mathsf{h}.\Pi$ のチャレンジャー $\mathsf{h}.\mathcal{C}$ に送る．そして，チャレンジャー $\mathsf{h}.\mathcal{C}$ から公開パラメータ $\mathsf{h}.\mathsf{PP}$ を受け取る．次に，攻撃者 $\mathsf{h}.\mathcal{A}$ は $\mathsf{r}.\mathsf{Setup}(1^\lambda) \to (\mathsf{r}.\mathsf{PP}, \mathsf{r}.\mathsf{MSK})$ を実行し，$(\mathsf{r}.\mathsf{kgc}, \mathsf{r}.\mathsf{MSK})$ が入っているリスト $\mathtt{SKList}$ を作る．さらに，$(\mathsf{h}.\mathsf{ku}_1, \mathsf{r}.\mathsf{MSK}') \leftarrow \mathsf{KeyUp}(\mathsf{r}.\mathsf{PP}, \mathsf{t}_{\mathsf{cu}} = 1, \mathsf{r}.\mathsf{MSK}, \mathsf{RL}_1 = \emptyset)$ を実行し，公開パラメータ $\mathsf{PP} := (\mathsf{r}.\mathsf{PP}, \mathsf{h}.\mathsf{PP})$ と更新鍵 $\mathsf{ku}_1 := \mathsf{h}.\mathsf{ku}_1$ を攻撃者 $\mathcal{A}_1$ に送る．その後，攻撃者 $\mathsf{h}.\mathcal{A}$ は，攻撃者 $\mathcal{A}_2$ の秘密鍵漏洩クエリ・鍵失効 & 更新鍵クエリ・復号鍵漏洩クエリに適切に答えることができることを示す．攻撃者 $\mathsf{h}.\mathcal{A}$ は，RIBE のマスター秘密鍵 $\mathsf{r}.\mathsf{MSK}$ を持っているため，全ての id, t に対して自ら（DKER を持たない）元となる RIBE の鍵を作ることができる．そして，攻撃者 $\mathsf{h}.\mathcal{A}$ は，チャレンジャー $\mathsf{h}.\mathcal{C}$ にクエリを行いながら，攻撃者 $\mathcal{A}_2$ の秘密鍵生成クエリ・秘密鍵漏洩クエリ・復号鍵漏洩クエリに以下のように答える：

秘密鍵生成クエリ： 攻撃者 $\mathcal{A}_2$ のクエリ $\mathsf{id} \in \mathcal{ID}$ に対して，攻撃者 $\mathsf{h}.\mathcal{A}$ はチャレンジャー $\mathsf{h}.\mathcal{C}$ に id の第 1 階層秘密鍵生成クエリを行う．[*6]そして，$\mathsf{r}.\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{r}.\mathsf{GenSK}(\mathsf{r}.\mathsf{PP}, \mathsf{r}.\mathsf{MSK}, \mathsf{id})$ を自ら実行する．ただし，攻撃者 $\mathcal{A}_2$ には何も送らない．

秘密鍵漏洩クエリ： 攻撃者 $\mathcal{A}_2$ のクエリ $\mathsf{id} \in \mathcal{ID}$ に対して，攻撃者 $\mathsf{h}.\mathcal{A}$ はある $\mathsf{r}.\mathsf{sk}_{\mathsf{id}}$ に対して $(\mathsf{id}, \mathsf{r}.\mathsf{sk}_{\mathsf{id}}) \in \mathtt{SKList}$ が成り立つことを確認する．チャレンジャー $\mathsf{h}.\mathcal{C}$ に id の第 1 階層秘密鍵漏洩クエリを行い，$\mathsf{h}.\mathsf{sk}_{\mathsf{id}}$ を受け取る．秘密鍵 $\mathsf{sk}_{\mathsf{id}} = (\mathsf{r}.\mathsf{sk}_{\mathsf{id}}, \mathsf{h}.\mathsf{sk}_{\mathsf{id}})$ を攻撃者 $\mathcal{A}_2$ に送る．

復号鍵漏洩クエリ： 攻撃者 $\mathcal{A}$ のクエリ $(\mathsf{id}, \mathsf{t}) \in \mathcal{ID} \times \mathcal{T}$ に対して，攻撃者 $\mathsf{h}.\mathcal{A}$ は以下の条件が成り立つかを確認する：ある $\mathsf{r}.\mathsf{sk}_{\mathsf{id}}$ に対して $(\mathsf{id}, \mathsf{r}.\mathsf{sk}_{\mathsf{id}}) \in \mathtt{SKList}$，$\mathsf{t} \le \mathsf{t}_{\mathsf{cu}}$，$(\mathsf{id}, \mathsf{t}) \neq (\mathsf{id}^*, \mathsf{t}^*)$．そして，チャレンジャー $\mathsf{h}.\mathcal{C}$ に $(\mathsf{id}, \mathsf{t})$ の第 2 階層秘密鍵漏洩クエリを行い，$\mathsf{h}.\mathsf{sk}_{\mathsf{id},\mathsf{t}}$ を受け取る．$\mathsf{r}.\mathsf{GenDK}(\mathsf{r}.\mathsf{PP}, \mathsf{r}.\mathsf{sk}_{\mathsf{id}}, \mathsf{ku}_{\mathsf{t}})$ を実行し，復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}} := (\mathsf{r}.\mathsf{dk}_{\mathsf{id},\mathsf{t}}, \mathsf{h}.\mathsf{sk}_{\mathsf{id},\mathsf{t}})$ を攻撃者 $\mathcal{A}_2$ に送る．

上記のようにして，攻撃者 $\mathsf{h}.\mathcal{A}$ は攻撃者 $\mathcal{A}_2$ の全てのクエリに適切に答えられていることを確認する．まず，攻撃者 $\mathcal{A}_2$ による秘密鍵漏洩クエリは，$\mathsf{id} \in \mathcal{ID} \setminus \{\mathsf{id}^*\}$ を満たす．よって，攻撃者 $\mathsf{h}.\mathcal{A}$ は，チャレンジャー $\mathsf{h}.\mathcal{C}$ にクエリを行うことで，同じ id に対する全ての第 1 階層 HIBE 秘密鍵を得ることができ，攻撃者 $\mathcal{A}_2$ の秘密鍵漏洩クエリに適切に答えることができる．そして，攻撃者 $\mathcal{A}_2$ による秘密鍵漏洩クエリは，$(\mathsf{id}, \mathsf{t}) \neq (\mathsf{id}^*, \mathsf{t}^*)$ を満たす．よって，攻撃者 $\mathsf{h}.\mathcal{A}$ は，チャレンジャー $\mathsf{h}.\mathcal{C}$ にクエリを行うことで，同じ $(\mathsf{id}, \mathsf{t})$ に対する全ての第 2 階層 HIBE 秘密鍵を得ることができ，攻撃者 $\mathcal{A}_2$ の復号鍵漏洩クエリに適切に答えることができる．

攻撃者 $\mathcal{A}_2$ のチャレンジクエリへの返答は，攻撃者 $\mathcal{A}_1$ へのものと変わらないので割愛する．攻撃者 $\mathcal{A}_2$ のクエリ $(\mathsf{M}_0, \mathsf{M}_1)$ を受け取ると，チャレンジャー $\mathsf{h}.\mathcal{C}$ から $\mathsf{h}.\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}(\mathsf{M}_b \oplus \mathsf{r}.\mathsf{M})$ を受け取り，攻撃者 $\mathsf{h}.\mathcal{A}$ 自身が $\mathsf{r}.\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}(\mathsf{r}.\mathsf{M})$ を生成することで $\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}(\mathsf{M}_b) = (\mathsf{r}.\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}(\mathsf{r}.\mathsf{M}), \mathsf{h}.\mathsf{ct}_{\mathsf{id}^*,\mathsf{t}^*}(\mathsf{M}_b \oplus \mathsf{r}.\mathsf{M}))$ を攻撃者 $\mathcal{A}_2$ に送る．ゲームの最後に，攻撃者 $\mathcal{A}_2$ から b' 受け取ると，同じ b' をチャレンジャー $\mathsf{h}.\mathcal{C}$ に送りゲームを終了する．この構成によって，HIBE 方式 $\mathsf{h}.\Pi$ の selective 安全性を破る攻撃者 $\mathsf{h}.\mathcal{A}$ が構成でき，優位性が関係 $\mathsf{Adv}^{\mathtt{RIBE\text{-}sel}}_{\Pi,\mathcal{A}_2}(\lambda) = \mathsf{Adv}^{\mathtt{HIBE\text{-}sel}}_{\mathsf{h}.\Pi,\mathsf{h}.\mathcal{A}}(\lambda)$ を満たす． □

[*6] チャレンジャー $\mathsf{h}.\mathcal{C}$ は $\mathsf{h}.\mathsf{sk}_{\mathsf{id}}$ を生成するが，ここでは受け取らない．

4. DKER を持つ格子 RIBE

この章で，DKER を満たす格子に基づく初めての RIBE 方式を提案する．この構成は，DKER を持たない Chen らの格子に基づく弱 selective 安全な RIBE 方式 [5] と，Agrawal らの格子に基づく selective 安全な HIBE 方式 [1] を用い，第 3 章で提案した一般的構成に基づくものである．紙面の都合上，CS 法や格子暗号における基本的なアルゴリズムは割愛する．詳しくは [1], [4], [5], [6], [14], [16] などを参照されたい．以下のパラメータ n, m, α, α', q は安全性パラメータ 1^λ の関数であるとする．

$\mathsf{Setup}(1^\lambda) \to (\mathsf{PP}, \mathsf{MSK})$ ：安全性パラメータ 1^λ を入力とし，$i = 1, 2$ に対して $(\mathbf{A}_i, \mathbf{T}_{\mathbf{A}_i}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ を実行し，一様ランダムな行列 $\mathbf{B}_1, \mathbf{B}_2 \leftarrow \mathbb{Z}_q^{n \times m}$ と一様ランダムなベクトル $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ をサンプルし，最大 N 個の葉ノードを持つ CS 法の二分木 $\mathtt{BT}$ を生成し，公開パラメータ $\mathsf{PP} := (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_1, \mathbf{B}_2, \mathbf{u})$ とマスター秘密鍵 $\mathsf{MSK} := (\mathtt{BT}, \mathbf{T}_{\mathbf{A}_1}, \mathbf{T}_{\mathbf{A}_2}, \mathsf{RL}_{\mathsf{t}} = \emptyset)$ を出力する．ただし，平文空間は $\mathcal{M} = \{0, 1\}$，ID 空間 $\mathcal{ID} \subseteq \mathbb{Z}_q^n$，時刻空間 $\mathcal{T} \subseteq \mathbb{Z}_q^n$ とする．

$\mathsf{GenSK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}) \to (\mathsf{sk}_{\mathsf{id}}, \mathsf{MSK}')$ ： 公開パラメータ PP，マスター秘密鍵 MSK と $\mathsf{id} \in \mathcal{ID}$ を入力とし，二分木 $\mathtt{BT}$ の葉ノードの中からまだユーザーが割り当てられていないものをランダムに選び，id を割り当てその葉ノードを η_{id} とする．ノード $\theta \in \mathsf{Path}(\mathtt{BT}, \eta_{\mathsf{id}})$ に $\mathbf{u}_\theta$ が定義されているかを確認し，定義されていない場合は $\mathbf{u}_\theta \leftarrow \mathbb{Z}_q^n$ をサンプルし二分木 $\mathtt{BT}$ の情報を更新する．そして，各ノー

ド $\theta \in \mathsf{Path}(\mathsf{BT}, \eta_{\mathsf{id}})$ に対して，$\mathsf{SampleLeft}(\cdot)$ アルゴリズムを用いて $[\mathbf{A}_1|\mathbf{B}_1 + H(\mathsf{id})\mathbf{G}]\mathbf{e}_{\mathsf{id},\theta} = \mathbf{u}_\theta$ を満たすベクトル $\mathbf{e}_{\mathsf{id},\theta} \in \mathbb{Z}^{2m}$ を離散ガウス分布よりサンプルする．さらに，$\mathsf{ExtendLeft}(\cdot)$ アルゴリズムにより $\mathbf{T}_{[\mathbf{A}_2|\mathbf{B}_1+H(\mathsf{id})\mathbf{G}]}$ をサンプルし，秘密鍵 $\mathsf{sk}_{\mathsf{id}} = \left((\theta, \mathbf{e}_{\mathsf{id},\theta})_{\theta\in\mathsf{Path}(\mathsf{BT},\eta_{\mathsf{id}})}, \mathbf{T}_{[\mathbf{A}_2|\mathbf{B}_1+H(\mathsf{id})\mathbf{G}]}\right)$ と更新された MSK' を出力する．

$\mathsf{KeyUp}(\mathsf{PP}, \mathsf{t}, \mathsf{MSK}, \mathsf{RL}_\mathsf{t}) \to (\mathsf{ku}_\mathsf{t}, \mathsf{MSK}')$ ：公開パラメータ PP，時刻 t，マスター秘密鍵 MSK，鍵失効者リスト RL_t を入力とし，CS 法の KUNode アルゴリズムによってノード $\theta \in \mathsf{KUNode}(\mathsf{BT}, \mathsf{RL}_\mathsf{t})$ を得る．これらのノードに $\mathbf{u}_\theta$ が定義されているかを確認し，定義されていない場合は $\mathbf{u}_\theta \leftarrow \mathbb{Z}_q^n$ をサンプルし二分木 BT の情報を更新する．そして，各ノード $\theta \in \mathsf{KUNode}(\mathsf{BT}, \mathsf{RL}_\mathsf{t})$ に対して，$\mathsf{SampleLeft}(\cdot)$ アルゴリズムを用いて $[\mathbf{A}_1|\mathbf{B}_2 + H(\mathsf{t})\mathbf{G}]\mathbf{e}_{\mathsf{t},\theta} = \mathbf{u}_\theta$ を満たすベクトル $\mathbf{e}_{\mathsf{t},\theta} \in \mathbb{Z}^{2m}$ を離散ガウス分布よりサンプルし，時刻 t の更新鍵 $\mathsf{ku}_\mathsf{t} = (\theta, \mathbf{e}_{\mathsf{t},\theta})_{\theta\in\mathsf{KUNode}(\mathsf{BT},\mathsf{RL}_\mathsf{t})}$ と更新されたマスター秘密鍵 MSK' を出力する．

$\mathsf{GenDK}(\mathsf{PP}, \mathsf{sk}_{\mathsf{id}}, \mathsf{ku}_\mathsf{t}) \to \mathsf{dk}_{\mathsf{id},\mathsf{t}}$ or $\perp$ ：公開パラメータ PP，秘密鍵 $\mathsf{sk}_{\mathsf{id}}$ と更新鍵 ku_t を入力とし，これらに共通するノード θ を見つける．このようなノード θ がなければ $\perp$ を出力し，これは id の鍵がすでに失効していることを意味する．そうでなければ，ノード θ における秘密鍵 $\mathbf{e}_{\mathsf{id},\theta} = \left[\mathbf{e}^{\mathsf{L}}_{\mathsf{id},\theta}||\mathbf{e}^{\mathsf{R}}_{\mathsf{id},\theta}\right]$ と更新鍵 $\mathbf{e}_{\mathsf{t},\theta} = \left[\mathbf{e}^{\mathsf{L}}_{\mathsf{t},\theta}||\mathbf{e}^{\mathsf{R}}_{\mathsf{t},\theta}\right]$ をそれぞれ $\mathbb{Z}_q^m$ のベクトルの連結とみなし，$\mathbf{d}_{\mathsf{id},\mathsf{t}} = \left[\mathbf{e}^{\mathsf{L}}_{\mathsf{id},\theta} + \mathbf{e}^{\mathsf{L}}_{\mathsf{t},\theta}||\mathbf{e}^{\mathsf{R}}_{\mathsf{id},\theta}||\mathbf{e}^{\mathsf{R}}_{\mathsf{t},\theta}\right]$ を計算する．さらに，$\mathsf{SampleLeft}(\cdot)$ アルゴリズムを用いて方程式 $[\mathbf{A}_2|\mathbf{B}_1 + H(\mathsf{id})\mathbf{G}|\mathbf{B}_2 + H(\mathsf{t})\mathbf{G}]\mathbf{g}_{\mathsf{id},\mathsf{t}} = \mathbf{u}$ を満たすベクトル $\mathbf{g}_{\mathsf{id},\mathsf{t}} \in \mathbb{Z}^{3m}$ をサンプルし，$(\mathsf{id}, \mathsf{t})$ に対する復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}} = (\mathbf{d}_{\mathsf{id},\mathsf{t}}, \mathbf{g}_{\mathsf{id},\mathsf{t}})$ を出力する．

$\mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, \mathsf{M}) \to \mathsf{ct}_{\mathsf{id},\mathsf{t}}$ ：公開パラメータ PP，$(\mathsf{id}, \mathsf{t})$ と平文 M を入力とし，平文を $\mathsf{M} = \mathsf{r.M} \oplus \mathsf{h.M}$ という条件のもと $\mathsf{r.M}, \mathsf{h.M} \in \{0,1\}$ をランダムにサンプルする．$\mathbf{s}_1, \mathbf{s}_2 \leftarrow (\mathbb{Z}_q^n)^2$，$x_1, x_2 \leftarrow (D_{\mathbb{Z},\alpha q})^2, \mathbf{x}_1, \mathbf{x}_2 \leftarrow (D_{\mathbb{Z}^{3m},\alpha' q})^2$ をサンプルし，$c_0 = \mathbf{u}^\top\mathbf{s}_1 + x_1 + \mathsf{r.M}\left\lfloor\frac{q}{2}\right\rfloor$，$c_1 = \mathbf{u}^\top\mathbf{s}_2 + x_2 + \mathsf{h.M}\left\lfloor\frac{q}{2}\right\rfloor$，$\mathbf{c}_2 = [\mathbf{A}_1|\mathbf{B}_1 + H(\mathsf{id})\mathbf{G}|\mathbf{B}_2 + H(\mathsf{t})\mathbf{G}]^\top\mathbf{s}_1 + \mathbf{x}_1$，$\mathbf{c}_3 = [\mathbf{A}_2|\mathbf{B}_1 + H(\mathsf{id})\mathbf{G}|\mathbf{B}_2 + H(\mathsf{t})\mathbf{G}]^\top\mathbf{s}_2 + \mathbf{x}_2$ を計算し，暗号文 $\mathsf{ct}_{\mathsf{id},\mathsf{t}} := (c_0, c_1, \mathbf{c}_2, \mathbf{c}_3)$ を出力する．

$\mathsf{Decrypt}(\mathsf{PP}, \mathsf{dk}_{\mathsf{id},\mathsf{t}}, \mathsf{ct}_{\mathsf{id},\mathsf{t}}) \to \mathsf{M}'$ ：公開パラメータ PP，復号鍵 $\mathsf{dk}_{\mathsf{id},\mathsf{t}}$ と暗号文 $\mathsf{ct}_{\mathsf{id},\mathsf{t}}$ を入力とし，$c_0' = c_0 - \mathbf{d}_{\mathsf{id},\mathsf{t}}^\top\mathbf{c}_2 \in \mathbb{Z}_q$ と $c_1' = c_1 - \mathbf{g}_{\mathsf{id},\mathsf{t}}^\top\mathbf{c}_3 \in \mathbb{Z}_q$ を計算し，$|c_0' - \lfloor\frac{q}{2}\rfloor| < \lfloor\frac{q}{4}\rfloor$ ならば $\mathsf{r.M}' = 1$ を，そうでなければ $\mathsf{r.M}' = 0$ とする．同様に，$|c_1' - \lfloor\frac{q}{2}\rfloor| < \lfloor\frac{q}{4}\rfloor$ ならば $\mathsf{h.M}' = 1$ を，そうでなければ $\mathsf{h.M}' = 0$ とし，$\mathsf{M}' = \mathsf{r.M}' \oplus \mathsf{h.M}'$ を復号結果として出力する．

謝辞．本研究の一部は，JST CREST JPMJCR14D6 の支援および JSPS 科研費 17J05603 の助成を受けて行われた．

参考文献

[1] Agrawal, S., Boneh, D. and Boyen, X.: Efficient Lattice (H)IBE in the Standard Model, *EUROCRYPT 2010*, LNCS 6110, Springer, pp. 553–572 (2010).

[2] Boldyreva, A., Goyal, V. and Kumar, V.: Identity-based encryption with efficient revocation, *ACM CCS 2008*, ACM, pp. 417–426 (2008).

[3] Boneh, D. and Franklin, M. K.: Identity-Based Encryption from the Weil Pairing, *SIAM J. Comput.*, Vol. 32, No. 3, pp. 586–615 (2003).

[4] Cash, D., Hofheinz, D., Kiltz, E. and Peikert, C.: Bonsai Trees, or How to Delegate a Lattice Basis, *J. Cryptology*, Vol. 25, No. 4, pp. 601–639 (2012).

[5] Chen, J., Lim, H. W., Ling, S., Wang, H. and Nguyen, K.: Revocable Identity-Based Encryption from Lattices, *ACISP 2012*, LNCS 7372, Springer, pp. 390–403 (2012).

[6] Gentry, C., Peikert, C. and Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions, *STOC'08*, ACM, pp. 197–206 (2008).

[7] Ishida, Y., Watanabe, Y. and Shikata, J.: Constructions of CCA-Secure Revocable Identity-Based Encryption, *ACISP 2015*, LNCS 9144, Springer, pp. 174–191 (2015).

[8] Lee, K.: Revocable Hierarchical Identity-Based Encryption with Adaptive Security, *IACR Cryptology ePrint Archive*, Vol. 2016, p. 749 (2016).

[9] Lee, K., Lee, D. H. and Park, J. H.: Efficient Revocable Identity-Based Encryption via Subset Difference Methods, *IACR Cryptology ePrint Archive*, Vol. 2014, p. 132 (2014).

[10] Lee, K. and Park, S.: Revocable Hierarchical Identity-Based Encryption with Shorter Private Keys and Update Keys, *IACR Cryptology ePrint Archive*, Vol. 2016, p. 460 (2016).

[11] Libert, B. and Vergnaud, D.: Adaptive-ID Secure Revocable Identity-Based Encryption, *CT-RSA 2009*, LNCS 5473, Springer, pp. 1–15 (2009).

[12] Naor, D., Naor, M. and Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers, *CRYPTO 2001*, LNCS 2139, Springer, pp. 41–62 (2001).

[13] Seo, J. H. and Emura, K.: Revocable Identity-Based Encryption Revisited: Security Model and Construction, *PKC 2013*, LNCS 7778, Springer, pp. 216–234 (2013).

[14] Seo, J. H. and Emura, K.: Revocable Identity-Based Cryptosystem Revisited: Security Models and Constructions, *IEEE Trans. Information Forensics and Security*, Vol. 9, No. 7, pp. 1193–1205 (2014).

[15] Seo, J. H. and Emura, K.: Revocable hierarchical identity-based encryption via history-free approach, *Theor. Comput. Sci.*, Vol. 615, pp. 45–60 (2016).

[16] Takayasu, A. and Watanabe, Y.: Lattice-Based Revocable Identity-Based Encryption with Bounded Decryption Key Exposure Resistance, *ACISP 2017*, LNCS 10342, Springer, pp. 184–204 (2017).

[17] Watanabe, Y., Emura, K. and Seo, J. H.: New Revocable IBE in Prime-Order Groups: Adaptively Secure, Decryption Key Exposure Resistant, and with Short Public Parameters, *CT-RSA 2017*, LNCS 10159, Springer, pp. 432–449 (2017).